

HLA Time Management and DIS
or
Putting things in Order

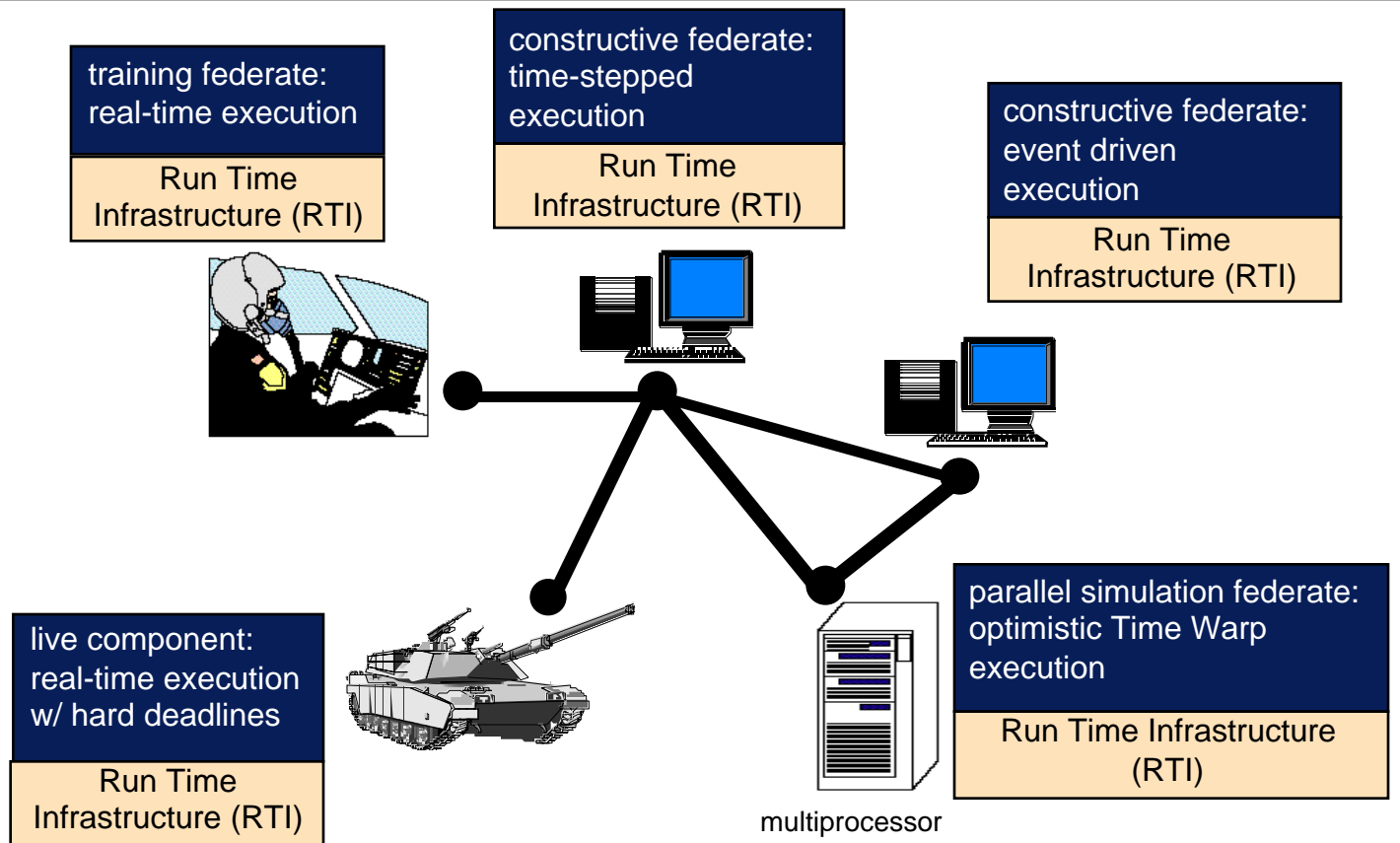
Richard M. Fujimoto
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
fujimoto@cc.gatech.edu

Richard M. Weatherly
The MITRE Corporation
7525 Colshire Drive
McLean, VA 22102-3481
weather@mitre.org

Outline

- Challenges
 - interoperability
 - latency vs. causality
- Message ordering
 - receive order
 - priority order
 - causal order
 - causal and totally ordered
 - time stamp order
- Summary / Why should I care about time management?

Challenge: Time Management Interoperability



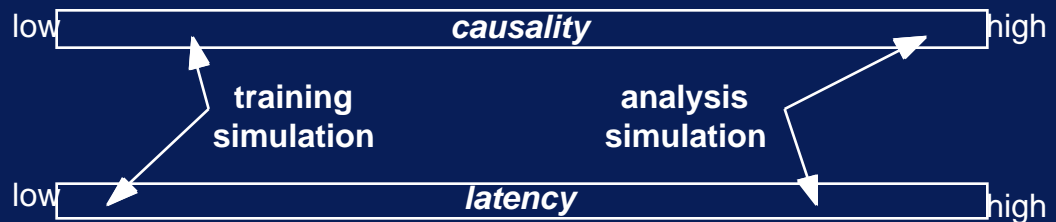
Goal: provide services to support interoperability among federates with different local time management schemes in a single federation execution.

Causality vs. Latency

The Challenge:

- different simulations have different requirements concerning “*causality*” and *latency*

Causality and Latency Requirements



- one time management structure must support interoperability among federates with different causality and latency requirements.

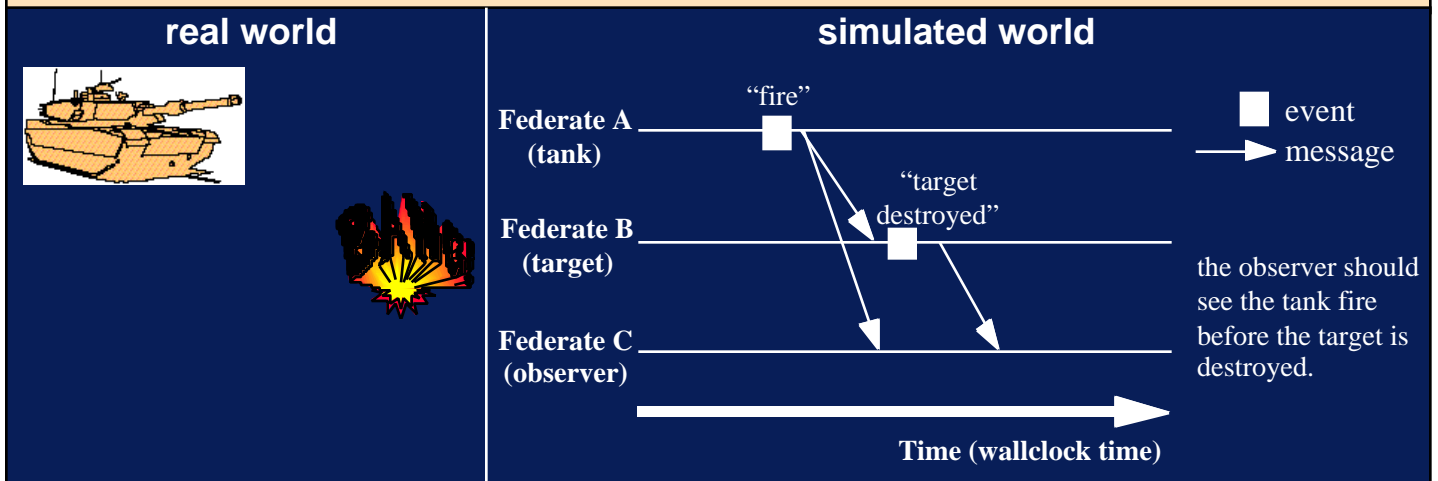
Approach:

The HLA defines a variety of service with different latency, and causality properties.

- Federates select the service(s) most appropriate for their requirements.
- Federates “get what they pay for.”

Causality

- “Things” happen in the real world in a certain order (e.g., cause & effect).
- It should appear that events in the simulated world happen in the same order as the real world actions that they represent.

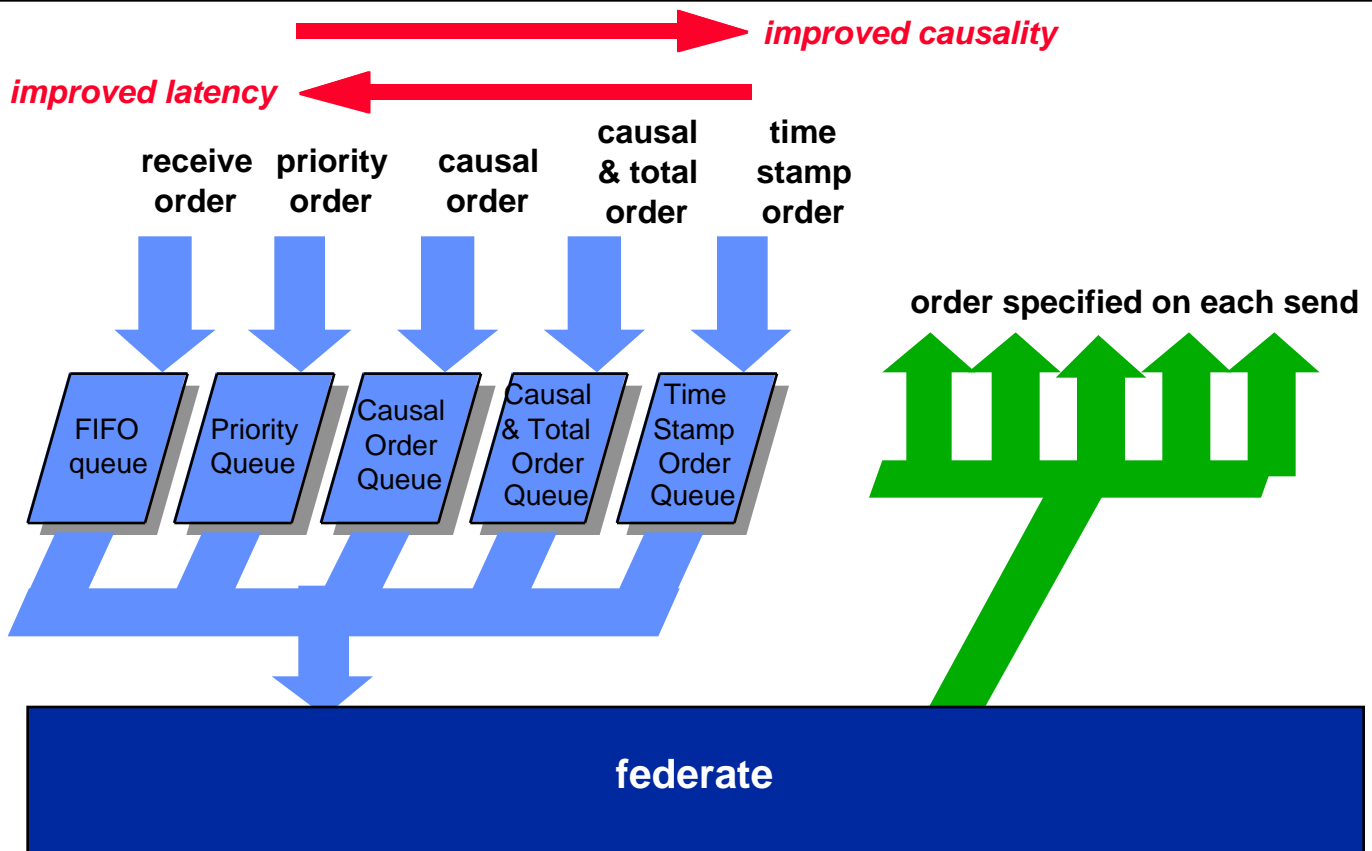


Observation: The key to producing causal distributed simulations is to ensure that messages are delivered to federates in the correct order.

Goal: If event A “happens before” event B, the message for A should be delivered before the message for B

HLA provides several levels of causality with different latency characteristics

Transportation Services



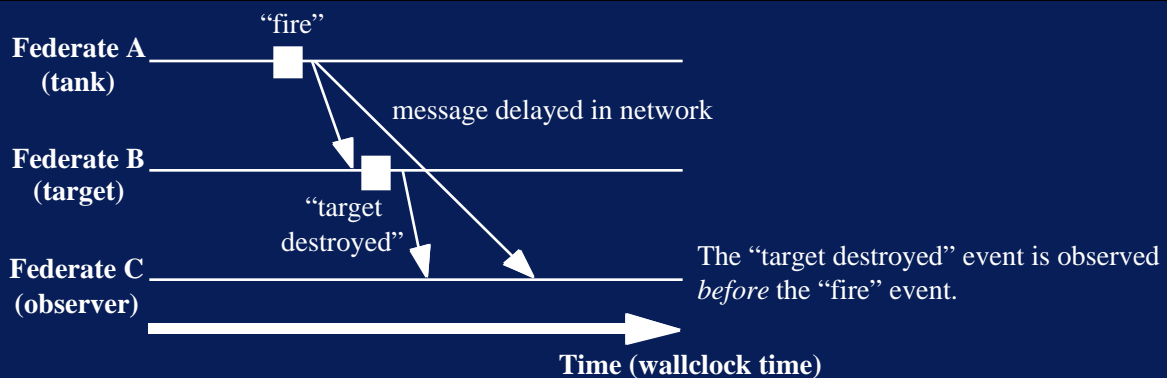
In addition to message ordering, transportation services also specify Quality of Service (reliable vs. best effort delivery)

Receive Order

latency 
causality 
low high

Receive Order

- incoming messages delivered to federate in the order they were received
- in general, **not** sufficient to create causal simulations



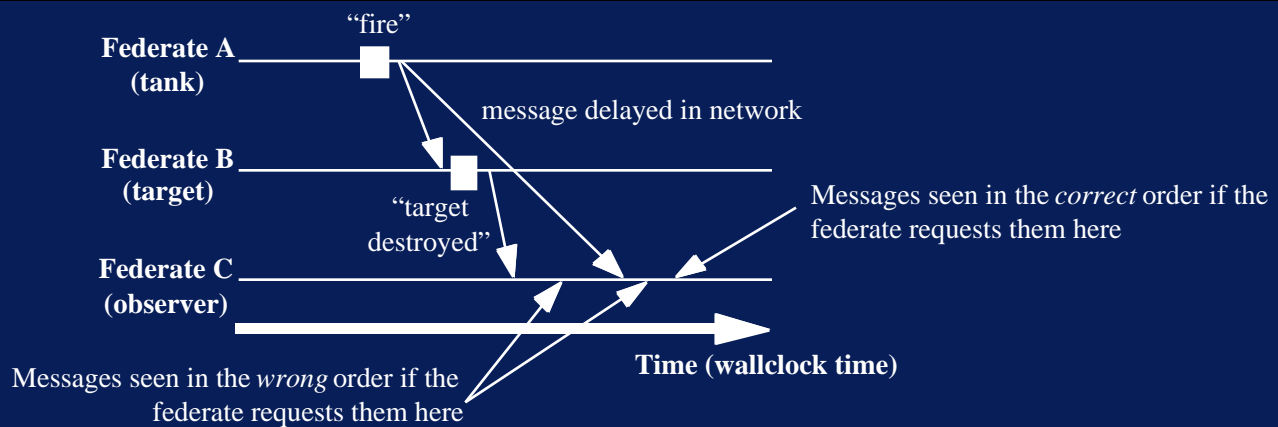
may get anomalies if time between causally related events is comparable to communication latencies (e.g., tightly coupled or scaled real-time simulations).

Receive order should be used if latency is primary concern, and some non-causal behavior can be tolerated.

Priority Order

<i>latency</i>	<div><div></div><div></div><div></div><div></div><div></div></div>
<i>causality</i>	<div><div></div><div></div><div></div><div></div><div></div></div>
	low high



- incoming messages stored in a priority queue with priority equal to the time stamp, deliver lowest time stamp first
- in general, **not** sufficient to create causal simulations



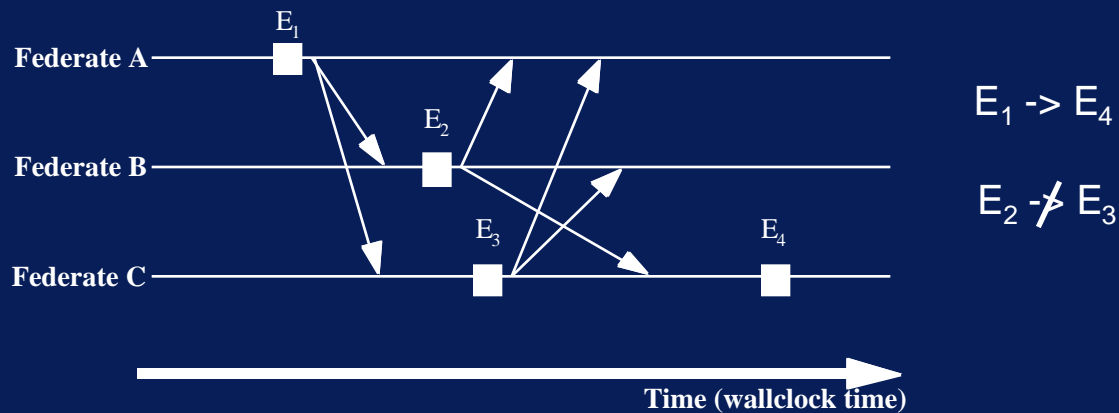
Messages are available for delivery as soon as they arrive.

Priority order should be used if latency is a primary concern, but some degree of ordering is considered desirable.

Causal Order

latency 
causality 
low high

- based on Lamport's "happens before" relationship (\rightarrow)
- if $E_1 \rightarrow E_2$, the message for E_1 will be delivered before the message for E_2





Each federate consists of an ordered sequence of actions, where an action is an (i) event, (ii) message send, or (iii) message receive. For two actions, A_1 and A_2 :

- if A_1 and A_2 occur in the same federate and A_1 precedes A_2 , then $A_1 \rightarrow A_2$
- if A_1 is a message send, and A_2 is a receive of the same message, then $A_1 \rightarrow A_2$
- if $A_1 \rightarrow A_2$ and $A_2 \rightarrow A_3$, then $A_1 \rightarrow A_3$ (transitivity)

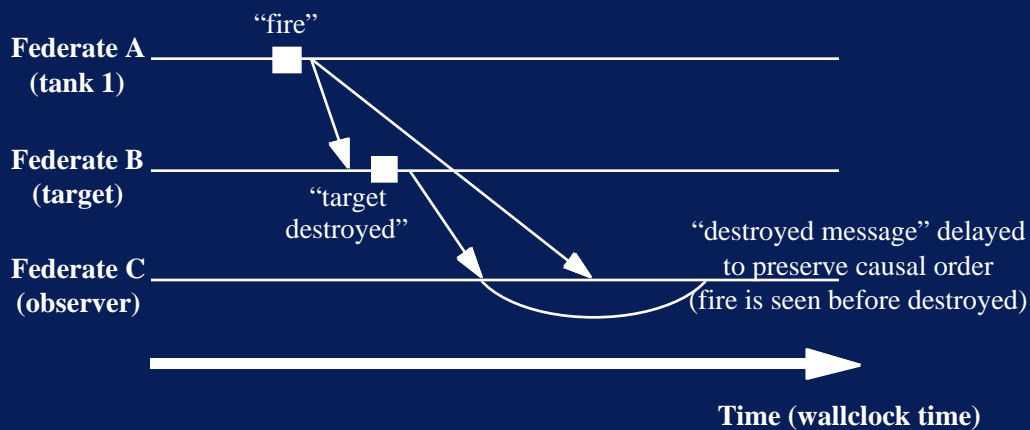
Actions that are not causally related are said to be **concurrent**.

Causal Order Message Delivery

latency 
causality 
low high

Causal Order



- if $E_1 \rightarrow E_2$, the message for E_1 will be delivered before the message for E_2
- messages for concurrent events may be delivered in any order; different federates may receive messages for concurrent events in different orders



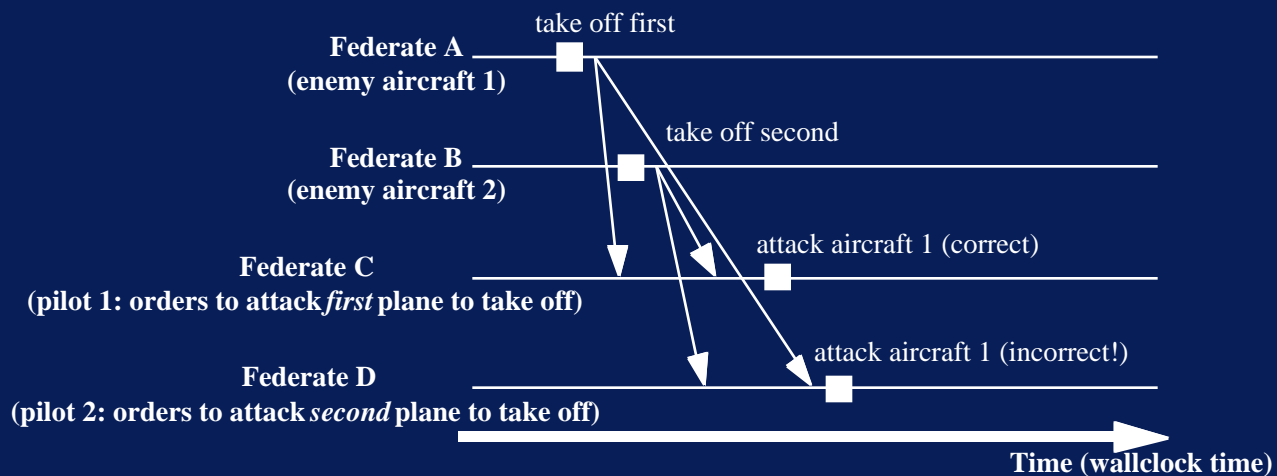
Messages may **not** be available for delivery as soon as they arrive.

Causal order should be used if some degree of causal guarantees are important, and some degree of latency increase can be tolerated.

Causal and Totally Ordered

latency 
causality 
low high

Observation: causal order may lead to certain anomalies



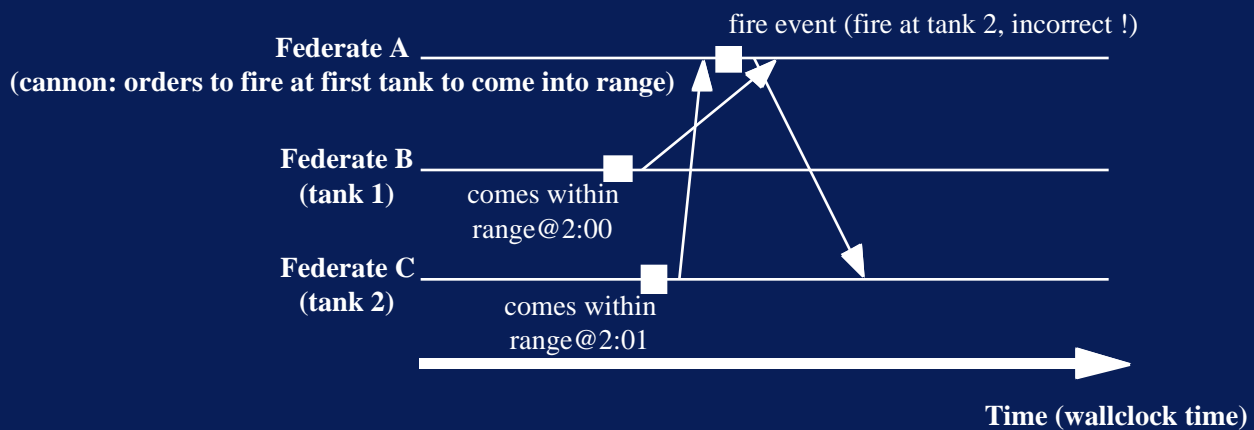
Causal and totally ordered communication service (CATOCS)

- provides causal order message delivery
- in addition, all federates receiving messages for the same events receive those messages ***in the same order.***

Causal and totally ordered should be used if consistent ordering of concurrent events is important.

Causal Order (and CATOCS): Limitations

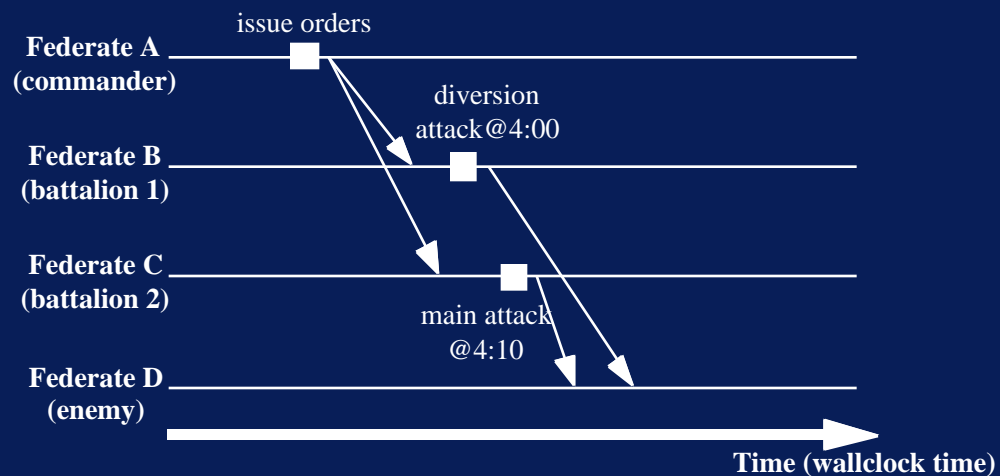
No ordering guarantees for concurrent events



- Federate A has orders to fire upon first target to come with range
- Federate B comes into range first, then Federate C comes into range
- "Come into range" events are concurrent; causal order does not guarantee any order of delivery
- B's message is delayed in the network; C's message is delivered to A first
- Cannon incorrectly fires upon C.

Causal Order (and CATOCS): Limitations

Hidden dependencies: dependencies between events that are not conveyed explicitly via messages may not be preserved.

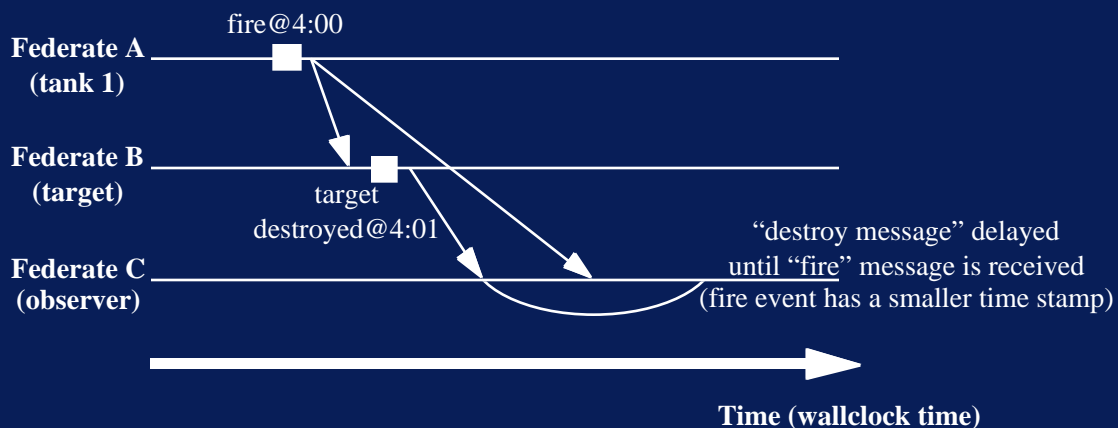


- Federate A issues orders for operation (diversion, then main attack)
- Federate B begins diversion attack
- Federate C begins main attack
- Messages from B and C are not causally related
- enemy federate observes the main attack before the diversion!

Time stamp Order

latency 
causality 
low high

- based on the **temporal happens before** relationship (\rightarrow_t):
- E_1 “happens before” E_2 ($E_1 \rightarrow_t E_2$) if E_1 has a smaller time stamp than E_2
- if $E_1 \rightarrow_t E_2$, the message for E_1 is delivered before the message for E_2



- eliminates all temporal anomalies and produces repeatable results
- requires “lookahead” (schedule events into the future) or optimistic message processing (e.g., Time Warp)

Time stamp order should be used if completely causal simulations are required (e.g., classical discrete event simulations).

HLA Time Management

- provides “time advance” services that prevent federates from receiving messages in their past
- allows different message ordering services to be used *within a single federate*
- supports inclusion of as-fast-as-possible and real-time simulations within a single federation execution*
- supports transparent inclusion of parallel simulations (even optimistic simulations)

* provided as-fast-as-possible simulation delivers real-time performance

HLA Time Management: Why should I (a DIS person) care?

- allows you to continue using receive order if that meets your needs
- allows you to add causality *for specific information* where this is needed (different levels of causality for different information in a single federate)
- supports federating DIS simulations with other simulations with stricter causality requirements (e.g., constructive simulations)

Message Ordering “consumer reports” summary *

latency causality

<i>receive order</i>	◐	●	◐ <i>as good as it gets</i>
<i>priority order</i>	◑	◒	◑ <i>pretty good</i>
<i>causal order</i>	○	○	○ <i>OK</i>
<i>CATOCS</i>	◒	◑	◒ <i>could be better</i>
<i>time stamp order</i>	●	◐	● <i>don't bet your job on it!</i>

* latency properties not yet fully determined, pending further experimentation

Initial implementation of RTI supports receive, priority, and time stamp order